

Lab Exercises

Lab Exercise 3 — Creating an Employee Class

Name: _____ Date: _____

Section: _____

This problem is intended to be solved in a closed-lab session with a teaching assistant or instructor present. The problem is divided into five parts:

1. Lab Objectives
2. Description of the Problem
3. Sample Output
4. Program Template (Fig. L 3.7, Fig. L 3.8 and Fig. L 3.9)
5. Problem-Solving Tips

The program template represents a complete working C++ program, with one or more key lines of code replaced with comments. Read the problem description and examine the sample output; then study the template code. Using the problem-solving tips as a guide, replace the `/* */` comments with C++ code. Compile and execute the program. Compare your output with the sample output provided. The source code for the template is available from the Companion Website for *C++ How to Program, Seventh Edition* at www.pearsonhighered.com/deitel/.

Lab Objectives

This lab was designed to reinforce programming concepts from Chapter 3 of *C++ How to Program, Seventh Edition*. In this lab, you will practice:

- Creating a class definition.
- Declaring data members.
- Defining a constructor.
- Defining *set* and *get* functions.
- Writing a test application to demonstrate the capabilities of another class.

Description of the Problem

Create a class called `Employee` that includes three pieces of information as data members—a first name (type `string`), a last name (type `string`) and a monthly salary (type `int`). [*Note:* In subsequent chapters, we'll use numbers that contain decimal points (e.g., 2.75)—called floating-point values—to represent dollar amounts.] Your class should have a constructor that initializes the three data members. Provide a *set* and a *get* function for each data member. If the monthly salary is not positive, set it to 0. Write a test program that demonstrates class `Employee`'s capabilities. Create two `Employee` objects and display each object's yearly salary. Then give each `Employee` a 10 percent raise and display each `Employee`'s yearly salary again.

Sample Output

```
Employee 1: Bob Jones; Yearly Salary: 34500
Employee 2: Susan Baker; Yearly Salary: 37800

Increasing employee salaries by 10%
Employee 1: Bob Jones; Yearly Salary: 37944
Employee 2: Susan Baker; Yearly Salary: 41580
```

Lab Exercises

Name: _____

Lab Exercise 3 — Creating an Employee Class

Program Template

```

1 // Lab 3: Employee.h
2 // Employee class definition.
3
4 #include <string> // program uses C++ standard string class
5 using namespace std;
6
7 // Employee class definition
8 class Employee
9 {
10 public:
11     /* Declare a constructor that has one parameter for each data member */
12     /* Declare a set method for the employee's first name */
13     /* Declare a get method for the employee's first name */
14     /* Declare a set method for the employee's last name */
15     /* Declare a get method for the employee's last name */
16     /* Declare a set method for the employee's monthly salary */
17     /* Declare a get method for the employee's monthly salary */
18 private:
19     /* Declare a string data member for the employee's first name */
20     /* Declare a string data member for the employee's last name */
21     /* Declare an int data member for the employee's monthly salary */
22 }; // end class Employee

```

Fig. L 3.7 | Employee.h.

```

1 // Lab 3: Employee.cpp
2 // Employee class member-function definitions.
3 #include <iostream>
4 using namespace std;
5
6 #include "Employee.h" // Employee class definition
7
8 /* Define the constructor. Assign each parameter value to the appropriate data
9    member. Write code that validates the value of salary to ensure that it is
10    not negative. */
11
12 /* Define a set function for the first name data member. */
13
14 /* Define a get function for the first name data member. */
15
16 /* Define a set function for the last name data member. */
17
18 /* Define a get function for the last name data member. */
19
20 /* Define a set function for the monthly salary data member. Write code
21    that validates the salary to ensure that it is not negative. */
22
23 /* Define a get function for the monthly salary data member. */

```

Fig. L 3.8 | Employee.cpp.

Lab Exercises

Name: _____

Lab Exercise 3 — Creating an Employee Class

```

1 // Lab 3: EmployeeTest.cpp
2 // Create and manipulate two Employee objects.
3 #include <iostream>
4 using namespace std;
5
6 #include "Employee.h" // include definition of class Employee
7
8 // function main begins program execution
9 int main()
10 {
11     /* Create two Employee objects and assign them to Employee variables. */
12
13     /* Output the first name, last name and salary for each Employee. */
14
15     /* Give each Employee a 10% raise. */
16
17     /* Output the first name, last name and salary of each Employee again. */
18 } // end main

```

Fig. L 3.9 | EmployeeTest.cpp .

Problem-Solving Tips

1. Class `Employee` should declare three data members.
2. The constructor must declare three parameters, one for each data member. The value for the salary should be validated to ensure it is not negative.
3. Declare a public *set* and *get* functions for each data member. The *set* functions should not return values and should each specify a parameter of a type that matches the corresponding data member (`string` for first name and last name, `int` for the salary). The *get* functions should receive no parameters and should specify a return type that matches the corresponding data member.
4. When you call the constructor from the `main` function, you must pass it three arguments that match the parameters declared by the constructor.
5. Giving each employee a raise will require a call to the *get* function for the salary to obtain the current salary and a call to the *set* function for the salary to specify the new salary.
6. Be sure to follow the spacing and indentation conventions mentioned in the text.
7. If you have any questions as you proceed, ask your lab instructor for help.